Incremental Learning of Hand Symbols Using Event-Based Cameras

Lungu, Iulia Alexandra ; Liu, Shih-Chii ; Delbruck, Tobi

Abstract: Conventional cameras create redundant output especially when the frame rate is high. Dynamic vision sensors (DVSs), on the other hand, generate asynchronous and sparse brightness change events only when an object in the field of view is in motion. Such event-based output can be processed as a 1D time sequence, or it can be converted to 2D frames that resemble conventional camera frames. Frames created, e.g., by accumulating a fixed number of events, can be used as input for conventional deep learning algorithms, thus upgrading existing computer vision pipelines through low-power, low-redundancy sensors. This paper describes a hand symbol recognition system that can quickly be trained to incrementally learn new symbols recorded with an event-based camera, without forgetting previously learned classes. By using the iCaRL incremental learning algorithm, we show that we can learn up to 16 new symbols using only 4000 samples for each symbol and achieving a final symbol accuracy of over 80%. The system achieves latency of under 0.5s and training requires 3 minutes for 5 epochs on an NVIDIA 1080TI GPU.

DOI: https://doi.org/10.1109/jetcas.2019.2951062

Incremental learning of hand symbols using event-based cameras

Iulia Alexandra Lungu, Shih-Chii Liu, and Tobi Delbruck
Sensors Group, Institute of Neuroinformatics, University of Zurich and ETH Zurich
Zurich, Switzerland
iulialexandra,shih,tobi@ini.uzh.ch

***Abstract*—Conventional cameras create redundant output especially when the frame rate is high. Dynamic vision sensors (DVSs), on the other hand, generate asynchronous and sparse brightness change events only when an object in the field of view is in motion. Such event-based output can be processed as a 1D time sequence, or it can be converted to 2D frames that resemble conventional camera frames. Frames created, e.g., by accumulating a fixed number of events, can be used as input for conventional deep learning algorithms, thus upgrading existing computer vision pipelines through low-power, low-redundancy sensors. This paper describes a hand symbol recognition system that can quickly be trained to incrementally learn new symbols recorded with an event-based camera, without forgetting previously learned classes. By using the iCaRL incremental learning algorithm, we show that we can learn up to 16 new symbols using only 4000 samples for each symbol and achieving a final symbol accuracy of over 80%. The system achieves latency of under 0.5s and training requires 3 minutes for 5 epochs on an NVIDIA 1080TI GPU.**

***Index Terms*—neuromorphic, event camera, deep networks, data-driven, incremental learning, robotics, convolutional networks**

I. Introduction

Convolutional neural networks (CNNs), a type of artificial neural network architecture, are frequently at the core of computer vision and artificial intelligence (AI). Many achievements in these fields are based on successful training of CNN architectures for various image classification tasks with images recorded using frame-based cameras. These cameras typically have redundant output, especially when produced at high frame rates. Event-based cameras such as the Dynamic Vision Sensor (DVS) [1], [2] operate akin to the transient pathway in biological eyes and asynchronously report only local brightness changes in the scene, thereby obviating the need to process entire images.

In a real-world scenario, it is sometimes useful to learn to recognize new objects or concepts. Unlike biological brains, artificial neural networks (ANNs) are not good at learning incrementally because of catastrophic forgetting [3]. Whenever new objects need to be classified, ANNs have to be trained on both old and new data. For low-resource platforms with limited computing power and storage, such an approach is not feasible. An incremental learning algorithm, iCaRL, proposed in [4], allows a trained network to learn new objects online, with only slow forgetting of the old ones. This learning is done by using distillation loss [4], [5] and prototype selection to better consolidate previous knowledge.

In an online scenario where a low-resource AI system has to quickly learn new objects, event-based cameras paired with the iCaRL algorithm are a powerful combination for incremental object recognition. In this paper, we present a hand symbol recognition system which uses an accumulation of a fixed number of DVS events [6], [7] to create sparse images of hand symbols for classification. iCaRL is used to quickly learn to recognize new symbols, while being capable of maintaining high classification accuracy for old symbols. In working towards our goal to enable continuous learning on low-power embedded hardware, this paper reports the reduction of resources required for training a network for new symbols in terms of time, computation and training data.

The outline of this paper is as follows: Section II presents the related literature (including our AICAS 2019 conference paper [8] on which this paper is based), and Section III introduces the methods used for incremental learning. Section IV reports a tradeoff analysis of accuracy versus architecture, training set size and training time. Section V summarizes the results and implications for future work.

II. Related work

In incremental learning, catastrophic forgetting occurs when an existing network is trained only with new data, thereby overwriting the old knowledge. Forgetting can be offset to a certain extent by various methods, such as exemplar storage of all classes (iCaRL) and knowledge distillation [4], [5]; elastic weight consolidation (EWC) [9]; and autoencoder or generative adversarial network (GAN) sample generation [10], [11]. EWC has been shown to not scale well for large networks and datasets [11], [12]. Using an autoencoder or GAN implies necessity of only storing a model instead of the class samples [10], [11]. Although the model storage memory is lower than the sample storage, generating the samples is much slower than reading them from memory.

This work builds on our rock-scissors-paper (RoShamBo) demonstration [7]. The demonstration plays the game of RoShamBo against a human opponent. Rather than trying to outguess the opponent, it gives the illusion this is occurring by quickly recognizing the human's symbol and showing the winning symbol in response. The symbol recognition is

This work was supported by Samsung under the Global Research *Neuromorphic Processor Project*.

performed by a CNN that is driven by frames of accumulated DVS events.

In this paper, we extend this previous work by adding continuous learning facilitated by the iCaRL algorithm. Our AICAS 2019 conference paper [8] described a prototype of this incremental learning system. Compared to [8], this paper extends the analysis of how iCaRL performs in different scenarios. In the conference paper, the results for only 5 and 100 training epochs were shown. Here, we complemented the previous work with results for a single epoch incremental training time, different amounts of base training data (*Restricted* vs. *Complete base*), as well as a shorter base training time (*Short base*). These analyses helped us better understand the limitations of iCaRL. We also included more details of how the demonstration worked and how the input from DVS was processed.

The updated system reported in this work is capable of recognizing new symbols shown by the human for less than a minute. In this time, a few hundred to a few thousand images are collected depending on how fast the person is moving their hand and thus how many events are generated. Training for the new symbols takes place immediately after the samples are collected, but as shown in Section IV, the time needed to successfully train two new symbols is less than 3 minutes. An additional improvement to the previous demonstration is that the symbol shown to users is the majority vote of the last 5 inferences. This simple filtering increases latency by about half a second, but it effectively removes outliers caused by noisy input samples.

III. Methods

In a live demonstration of an AI system learning to recognize human gestures, speed is of utmost importance. We used a DVS camera, the DAVIS240C, developed by us and commercialized by inilabs. Although the DAVIS outputs frames and events, we only needed to use the events for this work. This sensor provides dynamic range exceeding 100dB and minimizes delay to sub-millisecond values compared to conventional cameras. It also increases the sparsity of the input to the network, although here we did not exploit this sparsity for faster CNN processing. The sub-millisecond latency asynchronous brightness change events are accumulated into 64×64 2D pixel histograms of a constant number of events, referred to here as *constant-event* frames [6]. The event ON and OFF polarities are discarded because we are only interested in the shape outline of the symbols; this rectification also makes the demonstration work robustly on complex static background scenes. Since the frames are created using a constant count of events rather than a constant exposure duration, the frame rate is proportional to the hand speed. Slow hand movements generate constant-event frames at a low rate of about 2 Hz, while rapid hand movements generate frames at a higher rate of up to 500Hz. Our GPU can only process frames at about 7 Hz, but the software discards older data to always use the latest available data. By using constant-count frames, the DVS frame is not blurred even for the fastest hand movements, as it would be for a constant-duration frame.

In order to run this demonstration on a low-power device, we need to reduce not only the time it takes to record and classify the data, but also the time it takes to train new symbols, as well as the memory allocated for saving the old knowledge. In order to satisfy these constraints, we chose iCaRL [4] because it has bounded memory for old knowledge, it automates the choice of relevant information and it directly accesses this information instead of generating it on the fly.

We quantitatively compare 3 architectures of increasing size to see the effect on iCaRL accuracy. For most experiments we used ResNet-32, a residual network with 32 layers [13] that has 8.5M parameters and takes 190 MOp/frame to compute. For inference during the live demonstration, computing the ResNet-32 takes about 130ms per frame using TensorFlow 1.10.0 in CPU mode on a Linux PC running Ubuntu 18.04.

The iCaRL training consists of different phases: A *base* training phase and subsequent multiple *incremental* training phases. We can also think of the base training as the first phase in the incremental training process. In [4], there are more classes for the base training than for the subsequent incremental phases. In this work, the base knowledge consists of four classes, while each incremental phase adds two new symbols. The base training is performed on all available samples for each class and represents the core knowledge of the system. For each incremental phase, all the new data and only a subset of previous knowledge is used. iCaRL automatically selects some of the most representative samples (exemplars) for each of the classes previously encountered and only stores those in memory. After each incremental training phase, exemplars for the newly learned classes are stored. Each exemplar is chosen by maximizing the normalized dot product between the average feature vector over all the already chosen exemplars and the average feature vector of all the samples. The feature vector consists of the output of the penultimate layer of the ResNet.

A distillation loss [5] is used to better preserve knowledge about the known classes. By using a distillation procedure, the network is encouraged to output the same non-softmaxed scores (logits) for the old classes as it did in the previous training phases. This loss also ensures that backpropagation is applied to all output units for the old classes, since these scores all have non-zero values, unlike the one-hot labels for the new classes.

The final symbol classification is based on the same metric as the one used for exemplar selection: the distances between the feature vector associated with each sample and the mean feature vector of each set of class examplars are compared. The class corresponding to the minimum distance is selected. This selection method stands in contrast to the usual softmax output used for conventional classifier CNNs.

The incremental symbol learning demonstration is driven

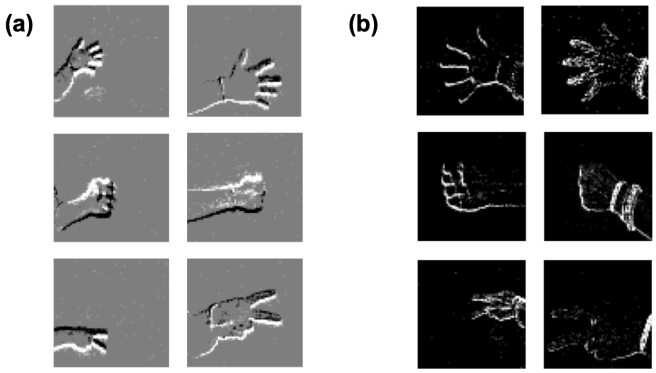

Fig. 1. (a): DVS samples before rectification and normalization. (b): DVS samples after rectification and normalization.

by a custom jAER class, RoShamBoIncremental[1] that communicates over a UDP socket to a python process. jAER [14] is a software project for processing data coming from event sensors. In our demonstration, it is used to capture the events, to turn them into frames and to display the output of the network. RoShamBoIncremental shows a GUI used to inform the system that a new symbol is being presented. Raw data is streamed to a file. When the demonstrator decides that a sufficient variety of data has been collected, they push a button that provides a dialog to enter the new class label. Next, RoShamBoIncremental sends a message to the python client to start incremental training on the new data. The python client produces the training dataset from the recorded data and starts training. When training is complete, RoShamBoIncremental loads the new CNN from disk.

Training data for this study consists of labeled continuous recordings of people showing a single symbol. The recordings are cut into frames using the jAER utility `dvs-slice-avi-writer`[2] and compiled into TensorFlow `tfRecords`. Input normalization (Fig. 1) includes the rectification of all DVS events to positive ON events with a 200-event maximum grayscale bin value and mapping the image pixel values to a 0-1 range by performing a 3-sigma normalization [6]. These normalization methods are applied both during training and at inference time. Figure 1 shows examples of the input classes before and after normalization. However, at inference time, jAER operates on individual frames as events are accumulated. It does not wait for an entire video to be recorded. The whole dataset consists of 20 distinct symbols, as shown in Fig. 2.

IV. Results

We performed 6 different types of experiments. The first one demonstrates catastrophic forgetting by comparing regular incremental training with iCaRL. The next 5 experiments are designed to evaluate the minimum resources needed to achieve over 80% classification accuracy at the end of a 8-phase incremental learning process. At the end of this process, our system can recognize 20 distinct hand symbols.

Sec IV-A first demonstrates the basic phenomenon of catastrophic forgetting. Then we test the number of base samples, the number of epochs for each incremental learning phase, as well as for the initial base training phase. Depending on the size of the dataset used during base training, we have two experiments: a *Complete base* reported in Sec. IV-B and a *Restricted base* scenario reported in Sec. IV-C. Each of these experiments tests different numbers of epochs of incremental training. In the Sec. IV-D *Short base* scenario, we also reduce the number of base training epochs. Sec. IV-E *Exemplars variation* experiments show how accuracy is influenced by the number of exemplars. All previous experiments were performed using a standard ResNet-32 architecture. To quantify the influence of the network parameter numbers on accuracy, we compared a 7-layer LeNet [15] architecture with two kinds of ResNet, a small and a large one. Sec. IV-F shows the effect of network size on accuracy.

The results are presented as the mean and standard deviation over 30 independent experiments. Before each experiment, the order in which the incremental classes arrive is shuffled.

A. Catastrophic forgetting experiment

We first illustrate the phenomenon of catastrophic forgetting. To this end, the accuracy of a network trained on incremental data in a regular fashion is compared to iCaRL. The regular network trains only on new data whenever it becomes available, without using distillation loss, exemplar saving or retraining on old data. Only one incremental phase is performed, because the accuracy for the regular network drops to zero on the previous classes as soon as the existing network is trained on new symbols.

For base training, we used a subset of the ROSHAMBO17 dataset[3], used for the original RoShamBo demonstration. ROSHAMBO17 consists of the three symbols used in the game, *rock*, *paper* and *scissors*, as well as a fourth *background* class which is intended to account for all other objects and backgrounds. Twenty users showed each symbol with each hand for about a minute and were instructed to explore all possible orientations, positions and scales. The background class data was collected from the DVS during no motion (noise events) and by showing the camera as many other scenes as possible, e.g. faces, body movements, and waving the camera around the office. All recordings for the base training were sampled using four different numbers of accumulated events: 500, 1000, 2000 and 4000. The resulting images were also mirrored to augment the data. This dataset has a total of 2.56 million images. The subset used for this analysis consisted of a total of 48k images, with 10k training images and 2k validation images for each base class.

Novel class data is acquired in the same manner as the base data, i.e., by accumulating events and cutting videos into frames. The recorded videos used in this study come from

[1]https://github.com/SensorsINI/jaer/blob/master/src/ch/unizh/ini/jaer/projects/npp/RoShamBoIncremental.java

[2]https://github.com/SensorsINI/jaer/blob/master/dvs-slice-avi-writer.sh]

[3]ROSHAMBO17: http://sensors.ini.uzh.ch/databases.html

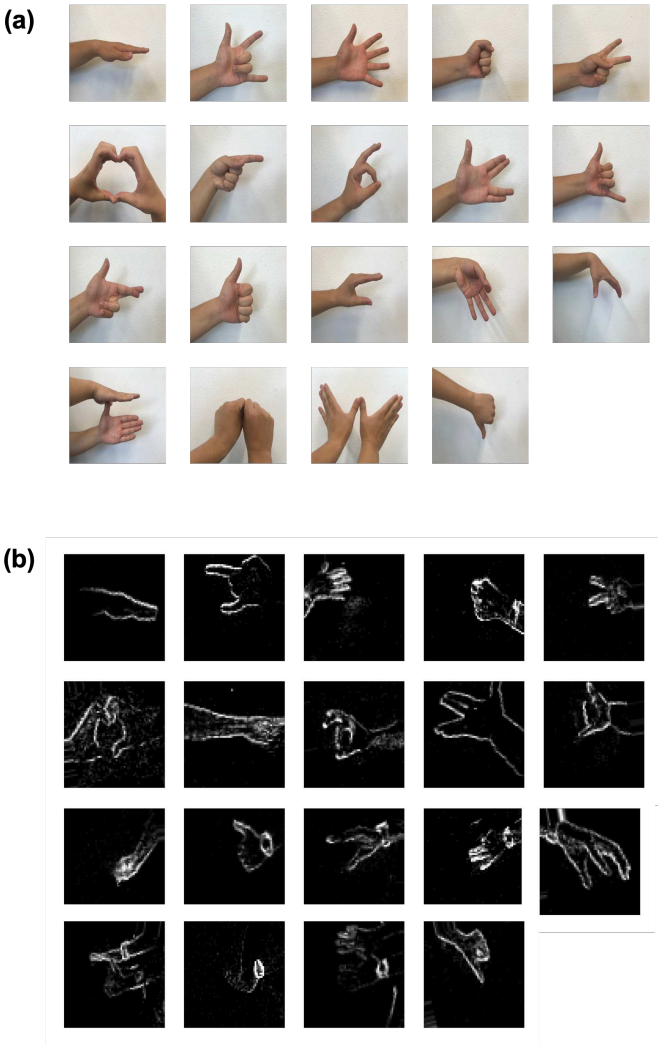

Fig. 2. All hand symbols in the dataset. Background class is omitted. (a) Hand symbols recorded with conventional camera. Not used in our experiments, for illustration purposes only. (b) DVS samples, used as input data for training iCaRL.

2 or 3 individuals depending on the particular hand symbol. The frames were pooled, shuffled and then divided so that 80% were used for training and 20% for validation. However, we used the demo paradigm for incremental learning of new symbols, that is, the videos recorded for each new class are much shorter than the base training data and are of unequal length, ranging from 15 seconds to 1 minute. Furthermore, we used only 2000-event DVS frames. To compensate for class imbalance and a lack of data, we augmented the new samples by random magnification, shear, and mirroring as many times as necessary to obtain 10k images for training and 2k images for validation for each new class. We saved 4k exemplars from each class for further incremental training.

Table I shows recognition accuracy results for base and incremental training after incremental learning of the first two new symbols. Both the regular and iCaRL networks are trained over 100 epochs reaching 99.4% accuracy on 4 base symbols ("BA after BT"). If the regular network is retrained (starting from the base weights) on only the samples for two new symbols, it achieves 99.56% accuracy on these two new symbols ("IA after IT"), but the accuracy on the original base classes drops to almost 0% ("BA after IT"). This extreme illustration of catastrophic forgetting shows that the network never selects the base classes, because it mistakes all samples as belonging to the new classes. Even after only a few epochs of training, the base class accuracy drops to less than 10%. However, if exemplars of all classes and distillation loss are used for the incremental training phase, the network maintains a high accuracy of 95.6% on the base classes while achieving an almost identical 95.7% accuracy on the two new classes. The results in Table I are based on 100 epochs of training.

TABLE I
REGULAR VERSUS INCREMENTAL TRAINING

Network type	BA after BT	BA after IT	IA after IT
Regular	99.4%	0.00002%	99.56%
iCaRL	99.4%	95.6%	95.7%

BA = accuracy on base classes
BT = training on base classes
IA = accuracy on incremental classes
IT = training incrementally on new classes

B. *Complete base*

For the *Complete base* experiments, we used the entire ROSHAMBO17 dataset as base knowledge and the same type of short demo videos described above for the incremental phases. Two new symbols were added at each incremental phase, to study the popular demo scenario where two subjects are each invited to contribute a new symbol before retraining.

Figure 3 shows how the accuracy evolves at each incremental phase, as well as how the number of incremental learning epochs we train on affects the performance of the system. The accuracy value corresponding to 4 classes is computed from the 4 base classes. Its standard deviation is zero, because the same base network was used for all 30 runs. At each incremental training stage we add 2 new classes. In total we learn 16 new symbols on top of the 4 base symbols. Different curves correspond to different training times. The base training phase is performed over 100 epochs in this scenario, as we expected the core knowledge to be particularly important for the network and we thus started from a well-trained baseline. However, for the incremental phases, we tested the impact of duration of training to minimise the time needed for retraining the network and therefore the waiting time of the participants.

Figure 3(a) shows the accuracy of the system on the latest learned classes. The mean accuracy tends to be stable over time. Figure 3(b) illustrates the drop in previous class recognition accuracy that happens over incremental phases: the more new classes iCaRL learns, the less accurate it becomes for the old classes. However, forgetting happens slowly, except for the case where each incremental phase consists of only 1 training epoch. Here we see a significant drop in accuracy

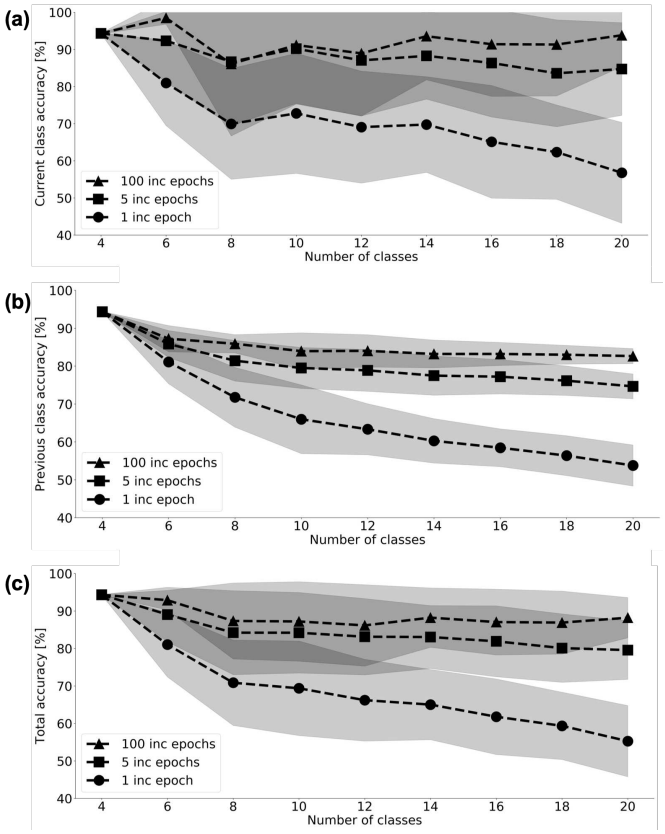

Fig. 3. Recognition accuracy of the ***Complete base*** iCaRL on different classes, for incremental training durations of 1, 5 or 100 epochs. The dotted curves represent the mean accuracy over 30 runs. The shading corresponds to the 1-sigma bounds. The new symbols added are shuffled at each run. (a) Accuracy on current classes (latest classes the algorithm was trained on). Tests how well new knowledge is acquired. (b) Accuracy on all previous classes after each incremental phase. Tests the level of old knowledge forgetting. (c) Overall recognition accuracy on old and current classes.

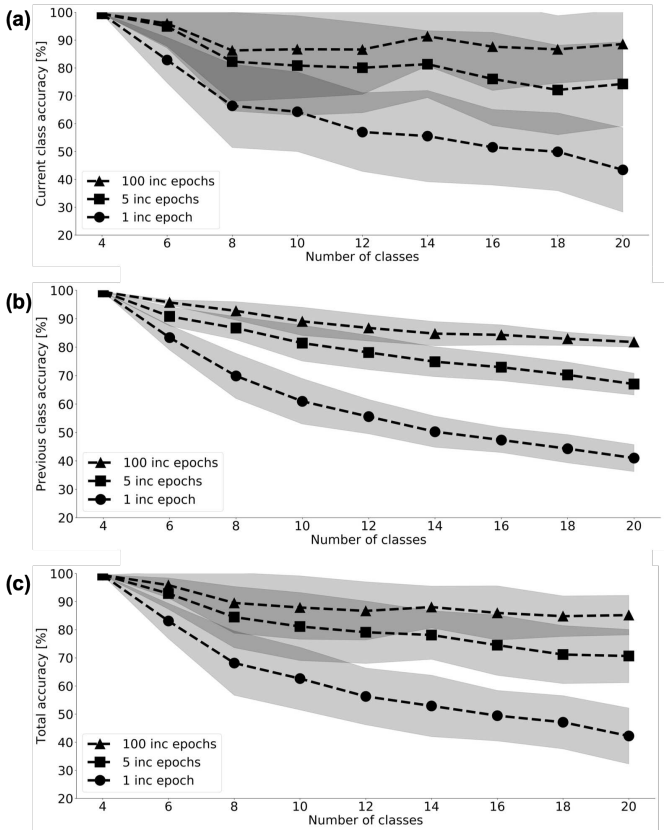

Fig. 4. Recognition accuracy of the ***restricted base*** iCaRL on different classes for incremental training durations of 1, 5 or 100 epochs. The dotted curves represent the mean accuracy over 30 runs. The shading corresponds to the 1-sigma bounds. The order in which new symbols are added is shuffled at each run. Experimental conditions for (a)-(c) are similar as in Fig. 3.

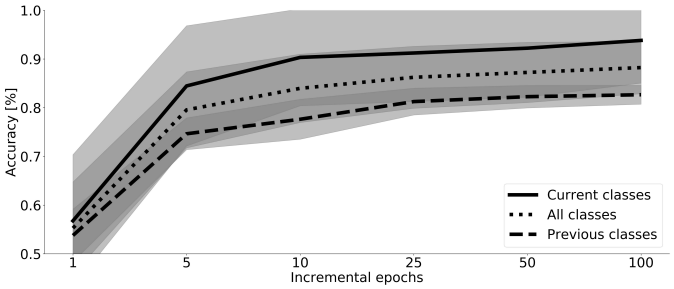

Fig. 5. Accuracy over incremental epochs in the *Complete base* scenario.

compared to the network that is trained for at least 5 epochs. Overall, the iCaRL algorithm helps to preserve very high recognition accuracy after 8 incremental training phases, even when trained over only 5 epochs (see Fig. 3(c)). The final total accuracies after 1, 5 and 100 epochs of incremental training are 55%, 80% and 89% respectively.

Accuracy variability results from a combination of random initial weights for the newly added output units, as well as the shuffling of symbols used for each incremental training phase. Some symbols are more difficult to distinguish than others.

Figure 5 shows the typical learning curves over epochs at the end of the last incremental learning stage, after having already learned 18 classes. These experiments were performed in the *Complete base* scenario. Learning happens very quickly after the first incremental epoch. The highest accuracy increase can be seen from epoch 1 to epoch 5. From these experiments, we concluded that 5 epochs of training for each incremental phase leads to satisfactory accuracy. Using 5 epochs also led to reduced training time during the live demonstration. For every two new symbols, training takes about 3 minutes on the NVIDIA GTX 1080 Ti GPU, in contrast to 15 minutes for 100 epochs.

C. Restricted base

For the *Restricted base* scenario, we did not use the entire 2.56M samples of the ROSHAMBO17 dataset for the base training. Instead, we randomly selected a subset of only 48k samples (1.8%), as explained in Sec. IV-A. The incremental training was performed as explained in the previous subsection. This scenario was included in order to quantify the dependence of iCaRL on the amount of base knowledge. Figure 4 shows the incremental accuracy is not drastically affected by a reduction in the number of base class samples. The final total

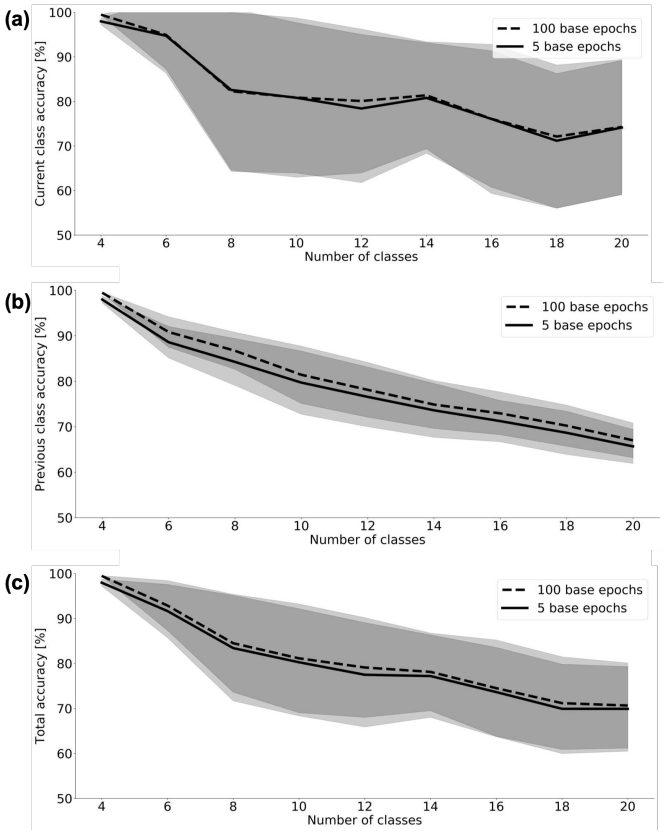

Fig. 6. Recognition accuracy for **short base training** with iCaRL. 5 and 100 epochs are compared for base training. 5 epochs were used for each incremental training phase. The dotted curves represent the mean accuracy over 30 runs. The shading corresponds to the 1-sigma bounds. The new symbols added are shuffled at each run.

accuracies after 1, 5 and 100 epochs of incremental training are 43%, 71% and 85% respectively. For an identical incremental training over 5 epochs, reducing the number of base samples, in this case from 640k to 10k samples per base class, results in an average accuracy reduction of 9% at the end of the 8-phase training process.

D. Short base training

In order to probe the limits of the algorithm, we tested how a reduced base training time affects the results. We trained over 5 epochs for the base training as well as for each incremental training phase, starting from a restricted base knowledge. In Fig. 6 the resulting accuracy is compared to that of experiments where we trained over 100 epochs during base training, but incrementally train over 5 epochs. The difference in accuracy between the two scenarios at the end of the whole training process is only 0.75%. It indicates that the dataset contains redundant information. The network learns to recognize the symbols even with only 5 epochs of training. Wall-clock training times are summarized in the next subsection.

E. Exemplars variation

In the original iCaRL paper, reducing the number of exemplars results in lower accuracy. For the following experiments, we tested 10, 100, 1000, 2000 and 4000 exemplars for each class. Figure 7 shows that reducing the number of exemplars per class reduces the overall accuracy. The final total average accuracy difference between using 4000 and 10 exemplars is 12%. The only number of exemplars that achieves the target 80 % accuracy at the end of the 9 phases is 4000. This scenario used the *Complete base* knowledge, 100 epochs base training and 5 epochs for incremental training.

Using 10 instead of 4000 exemplars saves us only 1 minute per incremental phase, but the drop in accuracy is significant. For this reason, we used 4000 exemplars per class for the live demonstration[4].

F. Architecture search

We tested 3 networks of different sizes in order to quantify the impact the number of parameters has on the accuracy of the algorithm. The simplest architecture is the 5-convolutional plus two fully connected layer LeNet [15] with 200k parameters. ResNet-32 has 8.5 million parameters, while ResNet-101 has 21 million parameters. Fig. 8 shows the results. After 8 incremental training phases, the average symbol recognition accuracy is 56.7% for LeNet, 70.64% for ResNet-32 and 72.30% for the larger ResNet-101. The smallest network clearly lacks capacity even to learn the base classes since the accuracy only reaches about 90%. Increasing the number of network parameters increases accuracy but only minimally after the network is large enough. The accuracy improvement by going from LeNet to Resnet-32 is more than 10%, but by going from ResNet-32 to ResNet-101 it is only 1.6%. These results show that our initial selection of Resnet-32 was satisfactory for the best tradeoff of inference time and accuracy. All experiments were performed on the restricted base scenario.

Overall, these experiments helped us reduce resource consumption for a live demonstration by choosing 5 epochs incremental training time, the ResNet-32 architecture and 4000 exemplars to store per class.

V. Conclusion

This paper introduced an event-driven hand symbol recognition system which can be incrementally trained to recognize new symbols without forgetting of old symbols even when the training is done in a few minutes. For each two new symbols, only 3 minutes are required for the incremental training phase. This corresponds to a factor of more than 100X faster training compared to the training time of a network which was retrained from scratch on all the data.

The memory needed for storing the training samples can also be reduced by a factor of 160X for the base training data, from 640k samples per class, to 4000 exemplars. The memory

[4]The video https://youtu.be/aWK572MMa1E shows the first live demonstration of our incremental learning system.

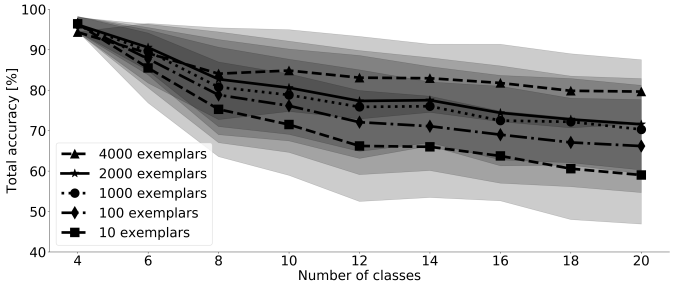

Fig. 7. Overall recognition accuracy of the incremental learning algorithm for different number of iCaRL exemplars. The shading corresponds to the 1-sigma bounds calculated over 30 experiments. The added new symbols are shuffled at each run

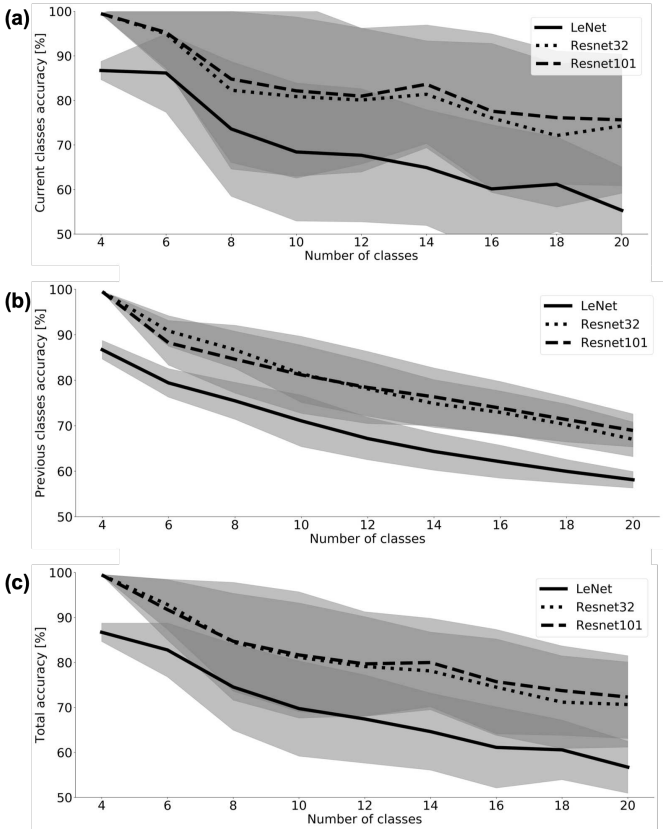

Fig. 8. Recognition accuracy of the incremental learning algorithm for different architectures in the restricted base scenario. The shading corresponds to the 1-sigma bounds calculated over 30 experiments. The added new symbols are shuffled at each run. (a) Accuracy on current classes (latest classes the algorithm was trained on). Tests how well new knowledge is acquired. (b) Accuracy on all previous classes after each incremental phase. Tests the retention of old knowledge. (c) Overall recognition accuracy on old and current classes.

for the incremental training data is also reduced by a factor of 2.5X, from 10k to 4k samples per class. The system can learn 16 new symbols for a total of 20 classes, and maintain overall single-sample recognition accuracy of 80% after 5 epochs of incremental learning, where chance level would be only 5%. Majority voting over the last 5 classifications filters out incorrect classifications and maintains latency to under half a second.

Using the minimal resources we established, a smartphone GPU with about 1% of the speed of a desktop GPU could compute the incremental training using about 64MB of training-set memory in about 5 hours training time. This is still a significant period but could take place during battery charging. Despite the contributions of the iCaRL algorithm to incremental learning, this algorithm still requires thousands of samples to learn new classes. To further reduce training time, we are looking into few-shot learning algorithms such as Siamese networks [16].

REFERENCES

[1] P. Lichtsteiner, C. Posch, and T. Delbruck, "A 128×128 120 dB 15μs latency asynchronous temporal contrast vision sensor," *IEEE Journal of Solid-State Circuits*, vol. 43, no. 2, pp. 566–576, Feb 2008.

[2] R. Berner, C. Brandli, M. Yang, S.-C. Liu, and T. Delbruck, "A 240× 180 10mW 12us latency sparse-output vision sensor for mobile applications," in *2013 Symposium on VLSI Circuits (VLSIC)*, 2013, pp. C186–C187.

[3] M. McCloskey and N. J. Cohen, "Catastrophic interference in connectionist networks : The sequential learning problem," in *Psychology of Learning and Motivation*, G. H. Bower, Ed. Academic Press, 1989, vol. 24, pp. 109–165. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0079742108605368

[4] S.-A. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert, "iCaRL: Incremental classifier and representation learning," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[5] G. E. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *CoRR*, vol. abs/1503.02531, 2015. [Online]. Available: http://arxiv.org/abs/1503.02531

[6] D. P. Moeys, F. Corradi, E. Kerr, P. Vance, G. Das, D. Neil, D. Kerr, and T. Delbruck, "Steering a predator robot using a mixed frame/event-driven convolutional neural network," in *2016 Second International Conference on Event-based Control, Communication, and Signal Processing (EBCCSP)*, June 2016, pp. 1–8.

[7] I. Lungu, F. Corradi, and T. Delbruck, "Live demonstration: Convolutional neural network driven by Dynamic Vision Sensor playing RoShamBo," in *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2017, pp. 1–1.

[8] I. A. Lungu, S. Liu, and T. Delbruck, "Fast event-driven incremental learning of hand symbols," in *2019 IEEE International Conference on Artificial Intelligence Circuits and Systems (AICAS)*, March 2019, pp. 25–28.

[9] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, D. Hassabis, C. Clopath, D. Kumaran, and R. Hadsell, "Overcoming catastrophic forgetting in neural networks," *Proceedings of the National Academy of Sciences*, vol. 114, no. 13, pp. 3521–3526, 2017. [Online]. Available: http://www.pnas.org/content/114/13/3521

[10] R. Kemker and C. Kanan, "FearNet: Brain-inspired model for incremental learning," in *6th Intl Conf on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018*, 2018. [Online]. Available: https://openreview.net/forum?id=SJ1Xmf-Rb

[11] A. Rios and L. Itti, "Closed-loop GAN for continual learning," *CoRR*, vol. abs/1811.01146, 2018. [Online]. Available: http://arxiv.org/abs/1811.01146

[12] R. Kemker, M. McClure, A. Abitino, T. L. Hayes, and C. Kanan, "Measuring catastrophic forgetting in neural networks," in *Proceedings of the 32nd AAAI Conference on Artificial Intelligence, (AAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, 2018, pp. 3390–3398. [Online]. Available: https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16410

[13] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[14] "jAER." [Online]. Available: http://jaerproject.org

[15] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov 1998.

[16] G. Koch, R. Zemel, and R. Salakhutdinov, "Siamese neural networks for one-shot image recognition," in *Proceedings of the 32nd International Conference on Machine Learning, Lille, France*, vol. 2, 2015.